

Instantiating and Monitoring Treatment Protocols

Serdar Uckun, MD, PhD

Rockwell International Science Center

444 High St., Palo Alto, CA 94301

ABSTRACT

This paper presents a system for protocol-based treatment planning, plan execution, and execution monitoring. The approach, named SPIN, is developed as a component of the Guardian system. Guardian is an experimental architecture for intelligent patient monitoring and control. The paper describes and illustrates SPIN in a clinical scenario.

1. INTRODUCTION

Protocol-based treatment plays a major role in critical care and emergency care. Since crisis situations are commonly encountered and are often associated with time pressure in these environments, there is usually no time for inventive thinking or decision making based on first principles. In a crisis situation, protocol-based treatment can ensure that all possible actions are considered by the clinician no matter how chaotic the situation. Examples are protocols for Advanced Cardiac Life Support (ACLS) [1] and anesthesia crisis management [2].

We are developing an intelligent agent architecture for patient monitoring and control applications named Guardian [3]. Guardian is designed to perform a variety of reasoning tasks in a critical care environment. These tasks range from data reduction and abstraction to higher-level cognitive skills such as diagnosis and therapy management. We recently developed a new therapy management component for Guardian which takes advantage of readily available treatment protocols by instantiating and executing skeletal treatment plans. In this article, we describe the method in detail, exemplify its use in a clinical scenario, and discuss the strengths and weaknesses of the representation and the execution framework.

2. METHODS

Our approach to therapy management is named SPIN, for *Skeletal Plan Instantiation*. In this approach, we define a *skeletal plan* as a hierarchical plan which outlines *all* management options for a given disorder. Each skeletal plan is recursively composed of finer-granularity plans. A hierarchy of plans terminates at actions which represent basic management steps. Skeletal plans and actions are instantiated at runtime according to the current context (i.e., patient status and the internal state of the agent). Instantiation in-

volves parameterization of actions and all execution decisions including branching. Figure 1 shows a conceptual hierarchy of treatment plans which may be individualized as a skeletal plan in SPIN.

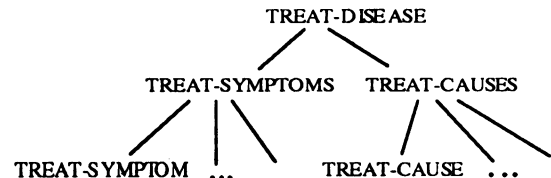


Figure 1. A conceptual hierarchy of treatment plans and actions.

2.1. Architecture

SPIN is a component of Guardian. However, rather than discussing Guardian in detail, we identify the functional requirements of a sufficient software environment for SPIN:

- A facility for continuous real-time monitoring of physical and physiological parameters, and abstraction and interpretation of these parameters into concepts such as clinical signs and diseases.
- A cache for all active plans and actions.
- An event-driven scheduler for plan execution monitoring (the "events" in question are either observations or changes in the status of active plans and actions).
- Another scheduler (ideally, with faster cycle time) for closed-loop control of actions.
- Three temporal databases to record interpreted values and planning activities (one timeline each for observations, intentions, and expectations).

Figure 2 illustrates the Guardian architecture from the perspective of planning skills.

2.2. Control Schemas

Each plan controls and monitors the execution of all its constituent plans and actions based on a control sentence named *control schema* (see Table 1). Plans are executed under the provision that control is local and hierarchical; that is, each plan can only control the execution of subplans and actions declared in a hierarchy rooted at that plan.

The simple syntax of the control schema is surprisingly powerful in capturing various orderings of ac-

tions and subplans under a master treatment plan. Further control on the execution of individual subplans and actions is exerted recursively by subplans and actions themselves.

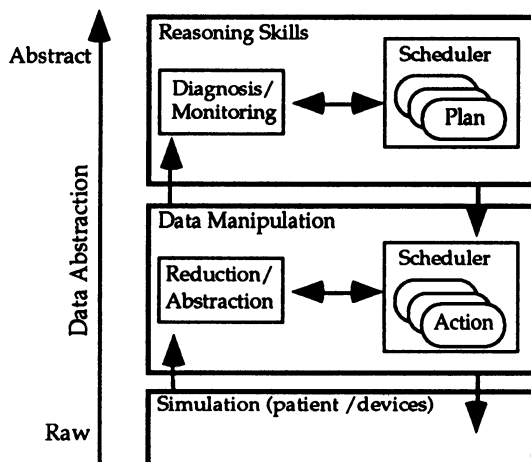


Figure 2. Planning in Guardian.

<pre> EXP ::= (op [plan action EXP]++) op ::= execute one-of all-of plan: pointer to another plan action: pointer to another action execute: all subsequent plans, actions, and expressions are to be executed in the provided sequence. one-of: only one of the listed plans, actions, or expressions is to be instantiated (in order of preference). all-of: all plans, actions, or expressions in this phrase are to be instantiated simultaneously. </pre>

Table 1. Syntax and semantics of control schemas.

2.3. Local Control of Plans and Actions

Each plan and action has three attributes which have Lisp expressions as their values. These expressions query the temporal database in order to assess whether certain conditions hold. The results of expression evaluation trigger state transitions for plans and actions.

- **preconditions** specify the conditions under which a plan or action can be executed. Preconditions are evaluated when a plan or action is in *standby* state; successful evaluation results in a transition to *active* state.
- **goal-conditions** specify the situations under which a plan or action may be terminated successfully. Thus, goal-conditions are only evaluated for active plans or actions. When evaluation is successful, a state transition to the *terminated* state takes place.

- **discontinuation-conditions** are conditions under which an active plan or action should be aborted before its goal conditions are satisfied. When these conditions hold, an active plan or action transitions to *discontinued* state.

Any state transition in a subplan causes its parent plan to transition to a *modified* state. Any plan which is in *modified* state reevaluates its control schema. This is how SPIN steps through active plans and performs plan execution monitoring.

2.4. Action Execution

The Guardian architecture supports two kinds of actions: support actions in the form of recommendations to the user, or closed-loop control actions¹. In order to equip an action with closed-loop control capability, two additional attributes need to be declared:

- **dosage-function**: a Lisp function which returns the new value of a controlled parameter based on current readings of related parameters.
- **control-interval**: specifies the period of the control loop, that is, how frequently the dosage-function should be reevaluated.

Three types of actions are possible in the closed-loop mode: actions that execute only once (e.g., one-time administration of a drug), actions that execute periodically (e.g., periodic administration of a drug), and continuous actions (e.g., controlling an infusion pump or O₂ administration). These categories only differ in the way they are handled by the action controller.

The action controller maintains a list of all active closed-loop actions. At each cycle (typically once every few seconds), the controller evaluates any action due for reevaluation based on its control-interval. Unless the goal-conditions or the discontinuation-conditions of a scheduled action is satisfied, the controller executes the dosage-function and propagates any prescribed changes to therapy parameters. If the action is terminated or discontinued, all active plans that relate to this action are notified. Subsequently, control schemas of all related plans are reevaluated.

3. MISMATCH RECOGNITION

SPIN does not make the assumptions that actions will be performed as requested, or that actions and plans will always result in desired or expected effects. Instead, it monitors the execution of plans and senses

¹Guardian is a proof-of-concept system which currently works on a simulated patient. We realize that significant issues such as legal aspects and safety have to be challenged and resolved before closed-loop control of therapy could become a reality in critical care.

any divergence from intentions and expectations and records its findings on three parallel timelines.

Information posted on the *observed* timeline includes actual parameter readings, values of clinical signs and symptoms, diagnostic hypotheses, and status information on treatment plans and actions. Intentions are the desired goals of treatment plans, and are posted on the *intended* timeline when a plan is activated. As one might expect, however, complete satisfaction of intentions is not a realistic expectation for every treatment plan. Some plans serve to remedy the adverse conditions slightly but cannot provide full recovery; others have significant side effects. Therefore, we record the expected effects of an activated plan in the *expected* timeline. The expectations of a plan are the sum of all expectations posted by its constituent actions. Figure 3 illustrates possible time courses of observations, intentions, and expectations for a hypothetical case of myocardial infarction. In this example, neither the treatment produce its expected results nor the intentions of the plan are realized. Subsequently, mismatch recognition may step in and attempt to reason about the situation. The mismatch recognition component of SPIN is under development.

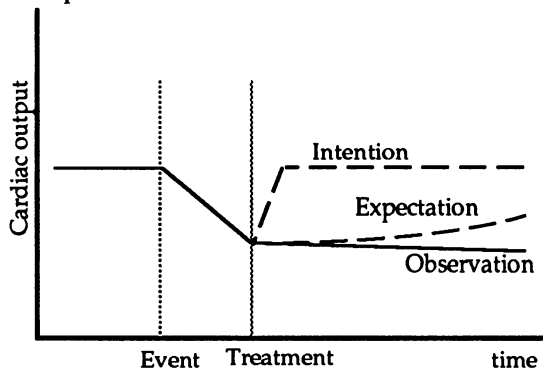


Figure 3. Intentions, expectations, and observations.

Observation-expectation mismatches may be used to 1) force plan revision in anticipation that a more effective treatment path might be chosen; 2) flag non-compliance to treatment and force reconsideration of relevant diagnostic hypotheses if all efforts fail; and 3) detect user compliance with treatment requests and to generate alerts when necessary. Inter-expectation mismatches may be used to identify potentially conflicting or redundant actions. These include actions prescribing different doses of the same drug, or actions neutralizing each other's effects. Intention-expectation mismatches may be used for plan optimization and revision by comparing the expectation-intention "distances" of possible treatment paths.

Similarly, observation-intention mismatches may be useful in situation assessment, i.e., determining the distance of the current patient state from a desired recovery state.

4. AN ILLUSTRATIVE EXAMPLE

We believe that SPIN offers considerable flexibility and strength in representing, executing, and monitoring treatment protocols. In this section, we exemplify the approach using a simple treatment scenario. Consider the following protocol:

In response to an acute bleeding episode, the clinician needs to monitor blood pressure, central venous pressure, heart rate, and hematocrit closely. Severe hypotension may be treated with IV bolus of vasopressor which may be repeated as necessary to maintain acceptable blood pressure. Blood volume should be restored using crystalloids. Blood transfusion should be used when blood or packed red blood cells are available, if bleeding cannot be controlled soon, and if hematocrit is too low (summarized and modified from [2]).

Table 2 illustrates part of the plan declaration for the management of bleeding. Table 3 shows the subplan for restoring blood volume, and Table 4 exemplifies a closed-loop control action which regulates the infusion of a crystalloid solution (normal saline). Due to space limitations, declarations as shown do not include all representational details.

We can represent vasopressor administration with a periodic closed-loop action. In this case, the control-interval will specify the frequency at which vasopressor administration should be reconsidered, the preconditions will specify the necessary conditions for repeated administration (possibly an assessment based on vital signs such as blood pressure and heart rate), and the dosage function will titrate the medication according to body weight and the intensity of desired effect.

The use of blood transfusion or packed red blood cells (RBCs) may be represented in a disjunctive phrase in the plan for restoring blood volume. In this case, both actions should be conditionalized on the availability of the related blood product and a low level of hematocrit. In addition, both preconditions may observe the duration of the bleeding episode and not authorize the action until the episode is long enough to warrant a transfusion. If both preconditions hold, packed RBCs will be preferred over whole blood according to the order specified in the ONE-OF phrase. Finally, both actions should have discontinuation conditions monitoring a transfusion reaction.

```
;; a plan to manage bleeding
;; CARDIAC is the name of the knowledge base
(def-bbl-object CARDIAC.manage-bleeding
  :attributes ((preconditions t)
               (control-schema
                 (ALL-OF CARDIAC.monitor-vital-signs
                          CARDIAC.manage-hypotension
                          CARDIAC.restore-blood-volume))))
```

Table 2. Partial declaration for a plan to manage bleeding.

```
;; a plan to restore blood volume
(def-bbl-object CARDIAC.restore-blood-volume
  :attributes ((preconditions t)
               (control-schema
                 (ALL-OF CARDIAC.infuse-normal-saline-high-rate
                          (ONE-OF CARDIAC.transfuse-packed-RBCs
                                   CARDIAC.transfuse-blood))))))
```

Table 3. Partial declaration for a plan to restore blood volume.

```
;; a closed-loop action to control fluid infusion
(def-bbl-object CARDIAC.infuse-normal-saline-high-rate
  :attributes ((preconditions t)
               (control-interval 60) ;; once every minute
               (dosage-fn
                 (let* ((weight (current-value-of-parameter 'body-weight))
                        (base-rate (weight * 1.0)) ;; ml/kg/min
                        (cvp (current-value-of-parameter 'central-venous-pressure)))
                   (cond ((> cvp 16) (* 0.8 base-rate))
                         ((< cvp 8) (* 1.6 base-rate))
                         ((< cvp 12) (* 1.2 base-rate))
                         (t base-rate))))
               (goal-conditions
                 (> (current-value-of-parameter 'mean-arterial-pressure) 90.0))
               (discontinuation-conditions nil)))
```

Table 4. Partial declaration for an action which controls an infusion rate in closed-loop (therapy specification is for illustration purposes only).

In the case of an adverse reaction, the discontinuation condition assures that the transfusion is aborted immediately.

When the overall treatment plan is instantiated, the action for a high-rate normal saline infusion will also be triggered. Since we expect an ICU patient to receive a saline infusion at all times, there will already be a previously-activated saline infusion action. In this case, SPIN will detect a potentially redundant instantiation and choose the high-rate infusion over the maintenance infusion. The maintenance infusion will be suspended until the high-rate infusion has accomplished its goals.

The failure of a plan step is not detrimental to the execution of SPIN. Assuming that the overall plan considers all possible outcomes of treatment, another remedial action will be chosen when the plan is reevaluated upon failure of one of its steps.

5. RELATED RESEARCH

Skeletal plan refinement was originally proposed by Friedland as a means to reduce the complexity of planning [4]. Similar ideas were exploited later in the PROTEAN [5] and PROTEGE/EON systems [6]. Instead of planning in an unconstrained search space, the skeletal plan refinement method relies on available abstract (or skeletal) plans which were refined in the context of a particular problem. SPIN further simplifies skeletal plan selection by caching top-level skeletal plans with each disorder. Thus, search is limited to local search and instantiation within a plan skeleton. In addition, SPIN merges plan instantiation and execution steps. As such, it integrates planning and replanning.

There are major differences between SPIN and traditional AI planners such as SIPE [7]. SPIN does not synthesize new plans using its knowledge of the domain. SPIN does not need to replan during execution either since all plan steps are already defined in the

protocol. In most classical planners, actions are instantaneous and execution follows a sequential thread. However, actions and plans are durative in SPIN and they are typically executed concurrently. As a consequence, SPIN actions and plans are interruptable and continuable since they execute over time intervals. SPIN also shares some features with intelligent agents which integrate high-level planners with low-level reactive controllers, such as PRS [8].

The DRIPS system uses abstraction hierarchies of actions selects optimal plans using a decision-analytic approach [9]. However, the selection depends on static features of the world and cannot be influenced by runtime events such as plan failure.

Finally, SPIN expands earlier medical AI efforts in protocol-based therapy such as ONCOCIN [10]. ONCOCIN is an expert system which supervises cancer chemotherapy protocols. It does not maintain state at runtime and its plans are difficult to maintain. In contrast, plans in SPIN are modular and easy to extract and represent in the form of protocols. In our experience, we had little difficulty representing treatment plans in SPIN even where protocols were not readily available in structured form.

6. DISCUSSION

6.1. Future Research

A major limitation in SPIN is in the control of plan and action selection. Disjunctive choices (ONE-OF) are resolved by trial-and-error in sequential order. This approach may be acceptable in an experimental setting, but in the real world of clinical medicine it is deficient. Future enhancements to SPIN should include decision-theoretic measures for selecting among conjunctive plans. These measures may include value, side effects, consequences, and cost. Reasoning about resource availability (e.g., personnel, devices) is another important target for SPIN.

6.2. Conclusions

In this paper, we present an architecture for skeletal plan instantiation, execution, and execution monitoring. This architecture operates in a highly uncertain environment where actions are durative and goal satisfaction is not always a reasonable expectation. Since actions may be taken on the basis of uncertain information, diagnosis tasks closely interact with treatment tasks and vice versa. Finally, plan construction is not only time consuming and difficult but also unnecessary in this domain since treatment protocols are already available in textbooks. Such features of the domain require a different approach to plan generation and execution than can be achieved

with classical AI planners. SPIN is the end result of these considerations.

We successfully used SPIN in the Guardian system on a number of simulated clinical scenarios. Further studies and improvements are underway. More rigorous assessments of the performance and representational adequacy of SPIN will be undertaken in a further study which involves comprehensive cognitive experiments.

Acknowledgments

This research was performed at the Knowledge Systems Laboratory at Stanford University. The Guardian project is sponsored by NASA contract NAG 2-581 under ARPA order 6822 and by a grant from the Whitaker Foundation administered by NSF. Barbara Hayes-Roth, David Gaba, John Drakopoulos, and Karl Pfleger provided helpful comments during the development of SPIN.

References

1. American Heart Association, *Textbook of Advanced Cardiac Life Support*. 1990, Dallas, TX.
2. Gaba, D.M., K.J. Fish, and S.K. Howard, *Crisis Management in Anesthesiology*. 1994, Churchill Livingstone, New York, NY.
3. Hayes-Roth, B., R. Washington, D. Ash, R. Hewett, et al., *Guardian: a prototype intelligent agent for intensive-care monitoring*. Artificial Intelligence in Medicine, 1992. 4 (2): 165-185.
4. Friedland, P.E., *Knowledge-based experiment design in molecular genetics*. PhD dissertation, Stanford University, 1979.
5. Hayes-Roth, B., B. Buchanan, O. Lichtarge, M. Hewett, et al. *PROTEAN: deriving protein structure from constraints*, in *Proc. AAAI-86*, Philadelphia, PA, pp. 904-909.
6. Musen, M.A., S.W. Tu, and Y. Shahar. *A problem-solving model for protocol-based care: from e-ONCOCIN to EON*, in *Proc. MEDINFO-92*. 1992. Geneva, Switzerland, pp. 519-525.
7. Wilkins, D.A., *Domain-independent planning: representation and plan generation*. Artificial Intelligence, 1984. 22: 269-301.
8. Georgeff, M.P. and A.L. Lansky. *Reactive reasoning and planning*, in *Proc. AAAI-87*. 1987. Seattle, WA: Morgan Kaufmann, pp. 677-682.
9. Haddawy, P. and M. Suwandi. *Decision-theoretic refinement planning using inheritance abstraction*, in *Proc. AIPS-94*. 1994. Chicago, IL: AAAI Press, pp. 266-271.
10. Langlotz, C.P., L.M. Fagan, S.W. Tu, B.I. Sikic, et al., *A therapy planning architecture that combines decision theory and artificial intelligence techniques*. Computers and Biomedical Research, 1987. 20: 279-303.